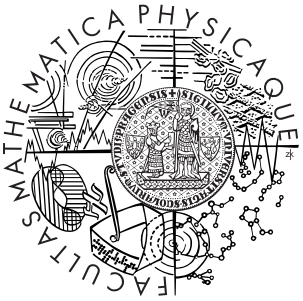
Univerzita Karlova v Praze
Matematicko-fyzikální fakulta

BAKALÁŘSKÁ PRÁCE



Miroslav Štufka

Particle filtering

Katedra pravděpodobnosti a matematické statistiky

Vedoucí bakalářské práce: Mgr. Blažena Frcalová

Studijní program: Matematika

Obor: Obecná matematika

2010

Rád bych na tomto místě vyjádřil poděkování především své vedoucí práce za cenné připomínky, rady a korektury a ochotu věnovat mi čas i ve chvílích svého volna. V neposlední řadě bych chtěl poděkovat i Matematicko-fyzikální fakultě UK za vzdělání, které mi během svého studia umožnila získat.

Prohlašuji, že jsem svou bakalářskou práci napsal samostatně a výhradně s použitím citovaných pramenů. Souhlasím se zapůjčováním práce a jejím zveřejňováním.

V Praze dne 14. 7. 2010 Miroslav Štufka

Obsah

Název práce: Particle filtering
Autor: Miroslav Štufka
e-mail autora: mirek@m-hosting.cz
Katedra (ústav): Katedra pravděpodobnosti a matematické statistiky
Vedoucí bakalářské práce: Mgr. Blažena Frcalová
e-mail vedoucího: blaza.frcalova@gmail.com

Abstrakt: Práce podává v první části úvod a definici skrytých Markovových modelů, poté vysvětluje jak fungují odhady metodou Monte Carlo. Dále se práce zabývá vylepšováním metody Monte Carlo, nejdříve generováním vzorků na základě důležitosti, potom sekvenčním generováním na základě důležitosti. Dále zavádí pojem přegenerovávání a kritérium efektivní velikosti vzorku. Jako shrnutí je představen obecný sekvenční algoritmus s adaptivním přegenerováváním. Tento aparát je poté použit na Particle filtering, kde je algoritmus názorně rozepsán po krocích. V druhé části práce je tento postup prakticky vyzkoušen na konkrétních nasimulovaných datech. Algoritmus je naprogramován v jazyce R. Na konci práce jsou shrnuty a prezentovány výsledky.

Klíčová slova: Bayesova věta, Skryté Markovovy modely, metoda Monte Carlo

Title: Particle filtering
Author: Miroslav Štufka
Author's e-mail address: mirek@m-hosting.cz
Department: Department of Probability and Mathematical Statistics
Supervisor: Mgr. Blažena Frcalová
Supervisor's e-mail address: blaza.frcalova@gmail.com

Abstract: In the first part of the thesis introduction and definition of Hidden Markov Models are given, then functioning of Monte Carlo methods for estimation are explained. Also, the thesis suggests an improvement of Monte Carlo method, first by Importance Sampling, next by Sequential Importance Sampling. Next, the term Resampling and Effective Sample Size criterion are defined. The previous is concluded in a Generic sequential algorithm with adaptive resampling. These tools are then used to Particle Filtering, where the algorithm is clearly described step by step. In the second part of the thesis, this algorithm is being tested on a specific simulated data. The algorithm is programmed in R language. At the end of the thesis, the results are explained and summarized.

Keywords: Bayes' theorem, Hidden Markov models, Monte Carlo method

Úvod

Cílem této bakalářské práce je přinést shrnutí základních teoretických výsledků o statistické metodě Particle filtering. Tyto výsledky poté odzkoušet na nasimulovaných datech.

Práce je rozdělena do čtyř kapitol. V první kapitole jsou zavedeny základní pojmy a věty, jejichž znalost v dalších částech práce předpokládáme. Ve druhé kapitole je popsán vlastní problém, který v práci řešíme. Postupně jsou vysvětleny skryté Markovovy modely a pojem filtrování. Dále je zde ukázán princip metody Monte Carlo, který je poté postupně vylepšován – nejdříve generováním vzorků na základě důležitosti, potom sekvenčním generováním na základě důležitosti. Na konci druhé kapitoly je zaveden pojem přegenerovávání a kritérium efektivní velikosti vzorku, díky němuž můžeme představit obecný sekvenční algoritmus Monte Carlo s adaptivním přegenerováváním. Ve třetí kapitole je tento získaný aparát aplikován na Particle filtering. Algoritmus je názorně zobrazen ve vývojovém diagramu s detailně rozepsanými kroky. Ve čtvrté kapitole jsou prezentovány výsledky praktického použití tohoto algoritmu na nasimulovaném modelu. Je zde k dispozici detailně komentovaný zdrojový kód v jazyce R a výstupy jsou zobrazeny v přehledných grafech. Za těmito čtyřmi kapitolami je zařazen závěr, který shrnuje přínos práce a možnosti rozšíření.

Součástí bakalářské práce je přiložené CD, které obsahuje text bakalářské práce ve formátu PDF a soubor se zdrojovým kódem simulace popsané ve čtvrté kapitole psaným v jazyce R.

Kapitola 1

Zavedení základních pojmů

V této sekci zavedeme několik definic a vět ([1], [2]), na které budeme v dalším textu odkazovat.

Pro jednoduchost zavedeme značení $z_{i:j} := (z_i, z_{i+1}, \ldots, z_j)$, pro libovolnou posloupnost $\{z_n\}_{n\geq 1}$ a jakákoliv přirozená $i \geq j$.

Definice 1. *Posloupnost náhodných veličin $\{X_n, n \in \mathbb{N}\}$ se nazývá Markovův řetězec s diskrétním časem a diskrétní množinou stavů χ, jestliže je splněna markovská vlastnost*

$$\mathrm{P}(X_{n+1} = y | X_n = x_n, X_{n-1} = x_{n-1}, \ldots, X_1 = x_1) = \mathrm{P}(X_{n+1} = y | X_n = x_n)$$

pro všechna $n \in \mathbb{N}$ a všechna $y, x_n, x_{n-1}, \ldots, x_1 \in \chi$ taková, že $\mathrm{P}(X_n = x_n, X_{n-1} = x_{n-1}, \ldots, X_1 = x_1) > 0$.

Nyní zavedeme Markovské jádro, které potřebujeme k definici Markovova řetězce s obecnou množinou stavů, stejně jako Pawlas v [1].

Definice 2. *Měřitelné zobrazení $P : \chi \times \xi \to [0,1]$ nazveme Markovským jádrem na (χ, ξ), pokud*

(i) pro každé $A \in \xi$ je $P(\cdot, A)$ nezáporná měřitelná funkce na χ,

(ii) pro každé $x \in \chi$ je $P(x, \cdot)$ pravděpodobnostní míra na ξ.

Definice 3. *Nechť P je Markovské jádro a ρ je pravděpodobnostní rozdělení, pak řekneme, že náhodný proces $\{X_n\}_{n\geq 1}$ s obecnou množinou stavů χ je homogenní Markovův řetězec s přechodovým jádrem P a počátečním rozdělením ρ, pokud jeho konečně rozměrná rozdělení splňují*

$$\mathrm{P}(X_1 \in A_1, \ldots, X_n \in A_n) = \int_{A_1} \cdots \int_{A_{n-1}} P(y_{n-1}, A_n) P(y_{n-2}, \mathrm{d}y_{n-1}) \ldots P(y_1, \mathrm{d}y_2) \rho(\mathrm{d}y_1)$$

pro každé $n \in \mathbb{N}$ *a pro všechna* $A_1, \ldots, A_n \in \xi$.

Definice 4. *Náhodný proces* $\{X_n, n \in \mathbb{N}\}$ *se nazývá gaussovský (normální), jsou-li všechna jeho konečně rozměrná rozdělení normální, tj. jestliže pro každé* $n \in \mathbb{N}$ *a* $t_1, \ldots, t_n \in \mathbb{N}$ *má vektor* $(X_{t_1}, \ldots, X_{t_n})'$ *n-rozměrné normální rozdělení* $\mathcal{N}_n(\boldsymbol{m_t}, \boldsymbol{V_t})$, *kde* $\boldsymbol{m_t} = (\mathrm{E}\, X_{t_1}, \ldots, \mathrm{E}\, X_{t_n})'$ *a*

$$\boldsymbol{V_t} = \begin{pmatrix} \operatorname{var} X_{t_1} & \operatorname{cov}(X_{t_1}, X_{t_2}) & \cdots & \operatorname{cov}(X_{t_1}, X_{t_n}) \\ \operatorname{cov}(X_{t_2}, X_{t_1}) & \operatorname{var} X_{t_2} & \cdots & \operatorname{cov}(X_{t_2}, X_{t_n}) \\ \vdots & \vdots & \ddots & \vdots \\ \operatorname{cov}(X_{t_n}, X_{t_1}) & \operatorname{cov}(X_{t_n}, X_{t_2}) & \cdots & \operatorname{var} X_{t_n} \end{pmatrix}$$

Věta 1 (**Bayesova**). *Nechť* f_X, f_Y *jsou pravděpodobnostní hustoty na* $\mathbb{R}^n$, *nechť* $y \in \mathbb{R}^n$, *potom pro všechna* $x \in \mathbb{R}^n$ *platí*

$$f_X(x|Y = y) = \frac{f_Y(y|X = x) f_X(x)}{\int_{-\infty}^{\infty} f_Y(y|X = \xi) f_X(\xi) \mathrm{d}\xi},$$

pokud je jmenovatel nenulový.

Definice 5. *Diracovo delta funkce* $\delta_{x_0}(x)$ *v bodě* $x_0 \in \mathbb{R}$ *je definována jako*

$$\delta_{x_0}(x) = \begin{cases} +\infty & \text{pro } x = x_0; \\ 0 & \text{jinak.} \end{cases} \quad a \int_{-\infty}^{\infty} \delta_{x_0}(x)\, \mathrm{d}x = 1.$$

Definice 6. *Diracova míra* δ_x *na množině* X *je pro každou měřitelnou podmnožinu* A *množiny* X *definována jako*

$$\delta_x(A) = \begin{cases} 0, & x \notin A; \\ 1, & x \in A. \end{cases}$$

Definice 7. *Nechť* $X_1, X_2, \ldots$ *je posloupnost nezávislých stejně rozdělených náhodných veličin s oborem hodnot* χ *a pravděpodobnostní mírou* P. *Empirická míra* P_n *je náhodná míra definována pro měřitelné podmnožiny množiny* χ *jako*

$$\mathrm{P_n}(A) = \frac{1}{n} \sum_{i=1}^{n} I_A(X_i) = \frac{1}{n} \sum_{i=1}^{n} \delta_{X_i}(A),$$

kde I_A *je indikátor a* δ_{X_i} *je Diracova míra.*

Kapitola 2

Vymezení problému

2.1 Skryté Markovovy modely

V této práci se budeme zabývat signály modelovanými jako Markovovy nelineární negaussovské procesy [3]. Předpokládejme tedy Markovův řetězec $\{X_n\}_{n\geq 1}$ s diskrétním časem, počátečním rozdělením μ a množinou stavů χ takový, že

$$X_1 \sim \mu(x_1) \text{ a } X_n|(X_{n-1} = x_{n-1}) \sim f(x_n|x_{n-1}),$$

kde $\mu(x)$ je pravděpodobnostní hustota a $f(x|x')$ značí hustotu přechodu od x' k x.

Nás zajímá odhad $\{X_n\}_{n\geq 1}$, ale máme přístup jen k procesu $\{Y_n\}_{n\geq 1}$. Předpokládáme, že pozorování $\{Y_n\}_{n\geq 1}$ jsou při dané realizaci procesu $\{X_n\}_{n\geq 1}$ statisticky nezávislá a jejich hustoty jsou dány vztahem

$$Y_n|(X_n = x_n) \sim g(y_n|x_n).$$

Pro jednoduchost zmiňujeme jen homogenní případ stejně jako v [3], tedy hustoty přechodu a pozorování nejsou závislé na indexu n.

Těmto modelům říkáme Skryté Markovovy modely, *Hidden Markov models* – (HMM). Tato třída pokrývá mnoho reálných situací, se kterými se můžeme v praxi setkat.

Dále máme

$$p(x_{1:n}) = \mu(x_1) \prod_{k=2}^{n} f(x_k|x_{k-1}),$$

$$p(y_{1:n}|x_{1:n}) = \prod_{k=1}^{n} g(y_k|x_k).$$

Ve smyslu Bayesovy věty dostáváme vztah

$$p(x_{1:n}|y_{1:n}) = \frac{p(x_{1:n}, y_{1:n})}{p(y_{1:n})},$$

kde

$$p(x_{1:n}, y_{1:n}) = p(x_{1:n})p(y_{1:n}|x_{1:n})$$

$$\text{a } p(y_{1:n}) = \int p(x_{1:n}, y_{1:n}) \mathrm{d}x_{1:n}. \tag{1}$$

Pro HMM s konečnou množinou stavů, integrál z (1) odpovídá konečné sumě a všechna tato diskrétní pravděpodobnostní rozdělení mohou být dopočtena. Některé další třídy modelů s dodatečnými předpoklady mohou být spočteny speciálními metodami, například Kalmanovými filtry v případě předpokladu normality. Nicméně pro většinu nelineárních negaussovských modelů není proveditelné spočítat tato rozdělení v uzavřeném tvaru a na řadě je tedy použití numerických metod. A právě Particle filtering, který je podtřídou velké skupiny sekvenčních metod typu Monte Carlo, *Sequential Monte Carlo* – (SMC), je souborem velmi silných metod na bázi simulace, které nám jsou schopny poskytovat odhady rozdělení $p(x_{1:n}|y_{1:n})$.

2.2 Filtrování a marginální věrohodnost

Filtrování je vlastně charakterizování rozdělení ve skrytých Markovových modelech v aktuálním čase, pokud jsou k dispozici všechna minulá pozorování (včetně současného). Dostáváme tedy

$$p(x_{1:n}, y_{1:n}) = p(x_{1:n-1}, y_{1:n-1}) f(x_n|x_{n-1}) g(y_n|x_n).$$

Následně dostáváme rekurzi

$$p(x_{1:n}|y_{1:n}) = p(x_{1:n-1}|y_{1:n-1}) \frac{f(x_n|x_{n-1}) g(y_n|x_n)}{p(y_n|y_{1:n-1})},$$

kde

$$p(y_n|y_{1:n-1}) = \int p(x_{n-1}|y_{1:n-1}) f(x_n|x_{n-1}) g(y_n|x_n) \mathrm{d}x_{n-1:n}. \tag{2}$$

Pro marginální rozdělení $p(x_n|y_{1:n})$ platí rekurze (3) – aktualizační krok a (4) – predikční krok.

$$p(x_n|y_{1:n}) = \frac{g(y_n|x_n) p(x_n|y_{1:n-1})}{p(y_n|y_{1:n-1})}, \tag{3}$$

kde

$$p(x_n|y_{1:n-1}) = \int f(x_n|x_{n-1}) p(x_{n-1}|y_{1:n-1}) \mathrm{d}x_{n-1}. \tag{4}$$

Pokud umíme postupně spočíst $\{p(x_{1:n}|y_{1:n})\}$ a tedy $\{p(x_n|y_{1:n})\}$, potom hodnotu marginální věrohodnosti $p(y_{1:n})$ můžeme rekurzivně spočíst pomocí

$$p(y_{1:n}) = p(y_1) \prod_{k=2}^{n} p(y_k|y_{1:k-1}),$$

kde $p(y_k|y_{1:k-1})$ je ze vzorce (2).

Bayesův vzorec v nelineárních a negaussovských dynamických modelech závisí na posloupnosti rozdělení $\{p(x_{1:n}|y_{1:n})\}$ a tedy $\{p(x_n|y_{1:n})\}$. Jak tvrdí Doucet a Johansen v [3], kromě jednoduchých případů není možné spočítat tato rozdělení v uzavřeném tvaru. V některých případech lze dosáhnout uspokojivých výsledků použitím aproximací funkcí těchto rozdělení, avšak v této práci se budeme zabývat jen aproximacemi metodou Monte Carlo. Hlavním přínosem této metody je, že pouze se slabými předpoklady přináší asymptoticky nestranné odhady požadovaných rozdělení. Navíc stojí za to poznamenat, že tato metoda může být použita na problémy vyšších dimenzí, v nichž tradiční numerické integrace dosahují slabších výsledků.

2.3 Metody typu Monte Carlo

Metody typu Monte Carlo se v matematice vyskytují velice často v různých souvislostech. Velmi zjednodušeně se dá říci, že se jedná o numerická schémata, kdy se zadaný problém aproximuje velkým množstvím náhodných jevů, které asymptoticky, při jejich počtu jdoucím k nekonečnu, dávají požadované výsledky. V této kapitole dále přiblížíme, jak se dá sekvenční metoda Monte Carlo, *Sequential Monte Carlo* – (SMC), použít na náš problém.

Sekvenční metoda Monte Carlo, je obecnou třídou metod Monte Carlo, která bere vzorky postupně (sekvenčně) z posloupnosti cílových pravděpodobnostních hustot $\{\pi_n(x_{1:n})\}$ zvyšující se dimenze, kde každé rozdělení $\pi_n(x_{1:n})$ je definováno na množině stavů χ^n (Protože $x_{1:n}$ je posloupnost n prvků množiny stavů χ).

Dále zavádíme funkci $\gamma_n : \chi^n \to \mathbb{R}^+$, u které požadujeme její bodovou znalost až na normalizační konstantu,

$$\pi_n(x_{1:n}) = \frac{\gamma_n(x_{1:n})}{Z_n}, \tag{5}$$

kde Z_n je normalizační konstanta zadaná

$$Z_n = \int \gamma_n(x_{1:n}) \mathrm{d}x_{1:n}, \tag{6}$$

která může být neznámá. SMC dává aproximaci $\pi_1(x_1)$ a odhad Z_1 v čase 1, potom aproximaci $\pi_2(x_{1:2})$ a odhad Z_2 v čase 2 a tak dále.

Nyní si představíme, jak vlastně metody typu Monte Carlo ve své podstatě fungují a následně doplníme některá vylepšení.

Předpokládejme aproximaci pravděpodobnostní hustoty $\pi_n(x_{1:n})$ pro pevné n. Vygenerujme N nezávislých náhodných vzorků, $X^i_{1:n} \sim \pi_n(x_{1:n})$ pro $i = 1 \dots N$, potom metoda Monte Carlo aproximuje $\pi_n(x_{1:n})$ pomocí empirické míry

$$\widehat{\pi}_n(x_{1:n}) = \frac{1}{N} \sum_{i=1}^{N} \delta_{X^i_{1:n}}(x_{1:n}), \tag{7}$$

kde $\delta_{x_0}(x)$ je Diracovo delta funkce $\delta_{x_0}(x)$ v bodě $x_0 \in \mathbb{R}$.

Na základě této aproximace můžeme aproximovat libovolnou marginální hustotu

$$\widehat{\pi}_n(x_k) = \frac{1}{N} \sum_{i=1}^{N} \delta_{X^i_k}(x_k),$$

a očekávanou hustotu libovolné funkce $\varphi_n : \chi^n \to \mathbb{R}$ dané

$$I_n(\varphi_n) := \int \varphi_n(x_{1:n})\pi_n(x_{1:n})\mathrm{d}x_{1:n},$$

odhadneme jako

$$I_n^{MC}(\varphi_n) := \int \varphi_n(x_{1:n})\widehat{\pi}_n(x_{1:n})\mathrm{d}x_{1:n} = \frac{1}{N}\sum_{i=1}^{N}\varphi_n(X_{1:n}^i).$$

Lze ověřit, jak je uvedeno v [2], že tento odhad je nestranný a jeho rozptyl je

$$\sigma^2_{I_n^{MC}(\varphi_n)} = \frac{1}{N}(\int \varphi_n^2(x_{1:n})\pi_n(x_{1:n})\mathrm{d}x_{1:n} - I_n^2(\varphi_n)).$$

To znamená, že se zvyšujícím se N se tento rozptyl snižuje.

2.4 Generování vzorků na základě důležitosti

V situacích, kdy z rozdělení $\pi_n(x_{1:n})$ neumíme vzorky generovat, což je třeba tehdy, když je rozdělení $\pi_n(x_{1:n})$ komplexní více dimenzionální, přichází na řadu generování vzorků na základě důležitosti, *Importance Sampling* – (IS). Jedná se o fundamentální a v praxi velmi často používané vylepšení (sekvenčně – viz další podkapitola). Principem IS je zavedení tzv. návrhové hustoty, *importance density*, $q_n(x_{1:n})$ tak, aby z ní šly vzorky generovat jednoduše a aby platilo

$$\pi_n(x_{1:n}) > 0 \Rightarrow q_n(x_{1:n}) > 0.$$

Ze vzorců (5) a (6) nám dává IS

$$\pi_n(x_{1:n}) = \frac{w_n(x_{1:n})q_n(x_{1:n})}{Z_n}, \tag{8}$$

$$Z_n = \int w_n(x_{1:n})q_n(x_{1:n})\mathrm{d}x_{1:n}, \tag{9}$$

kde $w_n(x_{1:n})$ je nenormovaná váhová funkce

$$w_n(x_{1:n}) = \frac{\gamma_n(x_{1:n})}{q_n(x_{1:n})}.$$

Generujme tedy N nezávislých vzorků $X_{1:n}^i \sim q_n(x_{1:n})$, kde $i = 1 \ldots N$. Tyto vzorky dosaďme do vzorce (7) pro základní variantu metody Monte Carlo

(Máme tedy aproximaci $\widehat{q}_n(x_{1:n})$ pomocí empirické míry vzorků $X^i_{1:n}$). Po dosazení do vzorců (8) a (9) dostáváme

$$\widehat{\pi}_n(x_{1:n}) = \sum_{i=1}^{N} W_n^i \delta_{X^i_{1:n}}(x_{1:n}),$$

kde

$$W_n^i = \frac{w_n(X^i_{1:n})}{\sum_{j=1}^{N} w_n(X^j_{1:n})}.$$

Očekávanou hodnotu funkce φ_n můžeme odhadnout jako

$$I_n^{IS}(\varphi_n) = \int \varphi_n(x_{1:n})\widehat{\pi}_n(x_{1:n})\mathrm{d}x_{1:n} = \sum_{i=1}^{N} W_n^i \varphi_n(X^i_{1:n}).$$

Na rozdíl od $I_n^{MC}(\varphi_n)$ není ovšem $I_n^{IS}(\varphi_n)$ nestranný pro konečná N, je ale konzistentní (uvedeno v [3]). V situacích, kdy umíme analyticky vyjádřit normalizační konstantu Z_n, můžeme sestrojit i odhad, který by byl nestranný, obecně by měl ale větší rozptyl.

2.5 Sekvenční generování vzorků na základě důležitosti

I když bychom uměli generovat vzorky z $q_n(x_{1:n})$, výpočetní náročnost takového modelu by neúměrně rostla s n. To znamená, že algoritmus, který by generoval vzorky postupně přímo z $q_n(x_{1:n})$, by měl vysokou složitost. Sekvenční generování vzorků na základě důležitosti, *Sequential Importance Sampling* – (SIS), přináší řešení tohoto problému volbou návrhové hustoty s následující vlastností

$$q_n(x_{1:n}) = q_{n-1}(x_{1:n-1})q_n(x_n|x_{1:n-1}) = q_1(x_1)\prod_{k=2}^{n} q_k(x_k|x_{1:k-1}).$$

To znamená, že k vygenerování vzorků $X^i_{1:n} \sim q_n(x_{1:n})$ v čase n, generujeme vzorky $X^i_1 \sim q_1(x_1)$ v čase 1, potom $X^i_k \sim q_k(x_k|X^i_{1:k-1})$ v časech $k = 2, \ldots, n$. Příslušné váhy lze počítat rekurzivně takto

$$w_n(x_{1:n}) = \frac{\gamma_n(x_{1:n})}{q_n(x_{1:n})} = \frac{\gamma_{n-1}(x_{1:n-1})}{q_{n-1}(x_{1:n-1})} \frac{\gamma_n(x_{1:n})}{\gamma_{n-1}(x_{1:n-1}) q_n(x_n|x_{1:n-1})},$$

což lze zapsat jako

$$\begin{aligned} w_n(x_{1:n}) &= w_{n-1}(x_{1:n-1})\alpha_n(x_{1:n}) \\ &= w_1(x_1) \prod_{k=2}^{n} \alpha_k(x_{1:k}), \end{aligned}$$

kde přírůstková váhová funkce důležitosti je

$$\alpha_n = \frac{\gamma_n(x_{1:n})}{\gamma_{n-1}(x_{1:n-1}) q_n(x_n|x_{1:n-1})}.$$

Rozumnou volbou $q_n(x_n|x_{1:n-1})$, která minimalizuje rozptyl $w_n(x_{1:n})$, je

$$q_n^{opt}(x_n|x_{1:n-1}) = \pi_n(x_n|x_{1:n-1})$$

a příslušná přírůstková váhová funkce důležitosti je potom

$$\alpha_n^{opt}(x_{1:n}) = \frac{\int \gamma_n(x_{1:n}) \mathrm{d}x_n}{\gamma_{n-1}(x_{1:n-1})} = \frac{\gamma_n(x_{1:n-1})}{\gamma_{n-1}(x_{1:n-1})}.$$

Ne vždy je ovšem možné generovat vzorky z $\pi_n(x_n|x_{1:n-1})$, v těchto případech můžeme návrhovou hustotu $q_n^{opt}(x_n|x_{1:n-1})$ aproximovat nebo zvážit jinou volbu.

I když budeme volit $q_n(x_n|x_{1:n-1})$ tak, aby z ní generování vzorků a následné počítání $\alpha_n(x_{1:n})$ bylo nezávislé na n, stále tato metoda ještě není dostačující.

2.6 Přegenerovávání vzorků

Toto vylepšení si klade za cíl vypouštět ty vzorky, jejichž váha už je příliš malá. Jinými slovy takové vzorky, že pravděpodobnost chování našeho procesu $\{X_n\}_{n\geq 1}$ podle nich už by byla příliš malá. Z tohoto intuitivně plyne, že nemůžeme přegenerovávání vzorků, *Resampling*, používat zase ani příliš často.

Základní myšlenka spočívá v uvědomění si, že aproximace založená na vážených vzorcích z $q_n(x_{1:n})$ nedává vzorky přibližně rozdělené vzhledem k $\pi_n(x_{1:n})$.

Napravit to můžeme právě přegenerováním vzorků z odhadu získaného SIS $\widehat{\pi}_n(x_{1:n})$. To provedeme tak, že vybereme $X^i_{1:n}$ s pravděpodobnostmi W^i_n. Tomuto kroku říkáme přegenerování, protože generuje vzorky z odhadu $\widehat{\pi}_n(x_{1:n})$, který už sám je získán generováním vzorků. Chceme-li tedy mít N vzorků získaných z $\widehat{\pi}_n(x_{1:n})$, přiřadíme každému původnímu vzorku $X^i_{1:n}$ počet jeho potomků N^i_n, $(i = 1 \dots N)$, způsobem, který si popíšeme za chvíli. Každému, tímto způsobem vzniklému, potomku přiřadíme váhu $\frac{1}{N}$. Vznikne nám tedy nový soubor $\{\frac{1}{N}, \overline{X}^i_n\}$ namísto starého $\{W^i_n, X^i_n\}$. Takže nakonec získáváme nový odhad $\widehat{\pi}_n(x_{1:n})$ pomocí přegenerované empirické míry

$$\overline{\pi}_n(x_{1:n}) = \sum_{i=1}^{N} \frac{N^i_n}{N} \delta_{X^i_{1:n}}(x_{1:n}).$$

Způsobů, jak získat počty potomků N^i_n, $(i = 1 \dots N)$, a zachovat nestrannost $\overline{\pi}_n(x_{1:n})$ je více (viz [2]). Nejhojněji používaný a jednoduše implementovatelný, jak tvrdí Doucet a Johansen v [3], je způsob systematického přegenerovávání, který funguje následovně.

Zvolme $U_1 \sim \mathcal{U}[0, \frac{1}{N}]$ a definujme $U_i = U_1 + \frac{i-1}{N}$ pro $i = 2, \dots, N$, nakonec položme

$$N^i_n = \left| \left\{ U_j : \sum_{k=1}^{i-1} W^k_n \leq U_j \leq \sum_{k=1}^{i} W^k_n \right\} \right|$$

s konvencí $\sum_{k=1}^{0} := 0$.

Přegenerovávání vzorků je vylepšením, které nám na jedné straně umožňuje generovat vzorky přímo z rozdělení $\pi_n(x_{1:n})$, na druhé straně zas přidává do systému dodatečnou chybu. Například při odhadování $I_n(\varphi_n)$ dosáhneme odhadu s nižším rozptylem při použití přímo odhadu $\widehat{\pi}_n(x_{1:n})$, namísto $\overline{\pi}_n(x_{1:n})$, (uvedeno v [3]). V každém případě přínosem přegenerovávání vzorků je odstranění případů s nízkou vahou. Jelikož budeme chtít aplikovat SMC na problematiku Particle filteringu, kde je důležité nezatěžovat výpočetní prostředky zbytečnými výpočty, zaměříme se raději na vzorky s vysokou vahou. Samozřejmě jdou zkonstruovat případy, kdy by bylo přegenerovávání na škodu, ale v běžných případech nám naopak zajišťuje stabilitu modelu v budoucnosti za cenu mírného zvýšení rozptylu v současnosti.

Poslední nezodpovězenou otázkou je, kdy tedy vzorky přegenerovávat? V jejím zodpovězení nám pomůže zavedení kritéria efektivní velikosti vzorku, *Effective Sample Size criterion* – (ESS).

Jelikož přegenerovávání přináší do systému dodatečný šum, je výhodnější používat ho jen ve chvílích, kdy je skutečně účinné. Když mají vzorky váhy s malým rozptylem, mohlo by být přegenerovávání nevhodné. V praxi je mnohem rozumnější přegenerovávat jen tehdy, když rozptyl vah je větší než předem zadaná tolerance. Toto realizujeme pomocí kritéria efektivní velikosti vzorku, které je zadáno v každém čase n jako

$$ESS = \left(\sum_{i=1}^{N} (W_n^i)^2 \right)^{-1}.$$

ESS nabývá hodnot od 1 do N a my přegenerováváme pouze tehdy, je-li pod hranicí N_T, typicky volenou jako $N_T = \frac{N}{2}$. Dostáváme tedy nový adaptivně přegenerovávaný odhad $\widehat{\pi}_n(x_{1:n})$

$$\widetilde{\pi}_n(x_{1:n}) = \sum_{i=1}^{N} W_n^i \delta_{X_{1:n}^i}(x_{1:n}).$$

2.7 Obecný sekvenční algoritmus Monte Carlo

Dáme-li tedy všechna tato fakta dohromady, dostáváme obecný sekvenční algoritmus Monte Carlo s adaptivním přegenerováváním. To znamená, že přegenerováváme jen tehdy, když už naše spektrum vzorků nepodává dostatečnou informaci o odhadovaném rozdělení. V prvním kroku spočteme odhad $\widehat{\pi}_1(x_1)$ dle vzorce (19). Pokud je poté $ESS < N_T$, provedeme přegenerování. Vyškrtáme tedy ty vzorky, které mají malou váhu – jinými slovy počet jejich potomků N_1^i je 0. Ostatní vzorky se ve výsledném souboru vyskytnou tolikrát, kolik vyšel jejich počet potomků N_1^i. Všem potomkům poté nastavíme stejné váhy $\frac{1}{N}$. Tím docílíme, že přegenerovaný soubor $\{\frac{1}{N}, \overline{X}_1^i\}$ je přibližně rozdělěn podle $\pi_1(x_1)$. Po tomto přegenerování se opět řídíme postupem SIS, to znamená, že generujeme $X_2^i \sim q_2(x_2|\overline{X}_1^i)$. Tedy $(\overline{X}_1^i, X_2^i)$ je přibližně rozdělen podle $\pi_1(x_1)q_2(x_2|x_1)$. Odteď můžeme příslušné váhy počítat pomocí přírůstkových vah důležitosti jednoduchým přenásobováním. Pokud je opět $ESS < N_T$, přegenerujeme vzorky vzhledem k normovaným vahám a takto pokračujeme dále.

Kapitola 3

Particle filtering

Vraťme se tedy k našemu původnímu problému, chceme počítat postupně rozdělení $\{p(x_{1:n}|y_{1:n})\}_{n\geq 1}$. Přímou aplikací obecného algoritmu SMC popsaného v minulé kapitole na cílová rozdělení $\pi_n(x_{1:n}) = p(x_{1:n}|y_{1:n})$ dostáváme požadovaný postup.

3.1 Algoritmus

Volme $\gamma_n(x_{1:n})$ z (5) jako $p(x_{1:n}, y_{1:n})$, čímž dostáváme $\pi_n(x_{1:n}) = p(x_{1:n}|y_{1:n})$ a $Z_n = p(y_{1:n})$. Nyní už nám stačí vhodně zvolit podmíněnou hustotu rozdělení důležitosti $q_n(x_n|x_{1:n-1})$. Volme tedy optimálně, jak tvrdí [3], abychom minimalizovali rozptyl vah. V čase n tedy máme $q_n^{opt}(x_n|x_{1:n-1}) = \pi_n(x_n|x_{1:n-1})$, kde

$$\pi_n(x_n|x_{1:n-1}) = p(x_n|y_n, x_n - 1) = \frac{g(y_n|x_n)f(x_n|x_{n-1})}{p(y_n|x_{n-1})}.$$

Příslušná přírůstková váhová funkce důležitosti je $\alpha_n(x_{1:n}) = p(y_n|x_{n-1})$.

V mnoha případech se ukazuje, že z tohoto rozdělení důležitosti nelze vzorky generovat, a proto běžně používáme tvar

$$q_n(x_n|x_{1:n-1}) = q(x_n|y_n, x_{n-1}).$$

V takovém případě dostáváme přírůstkovou váhovou funkci důležitosti takto

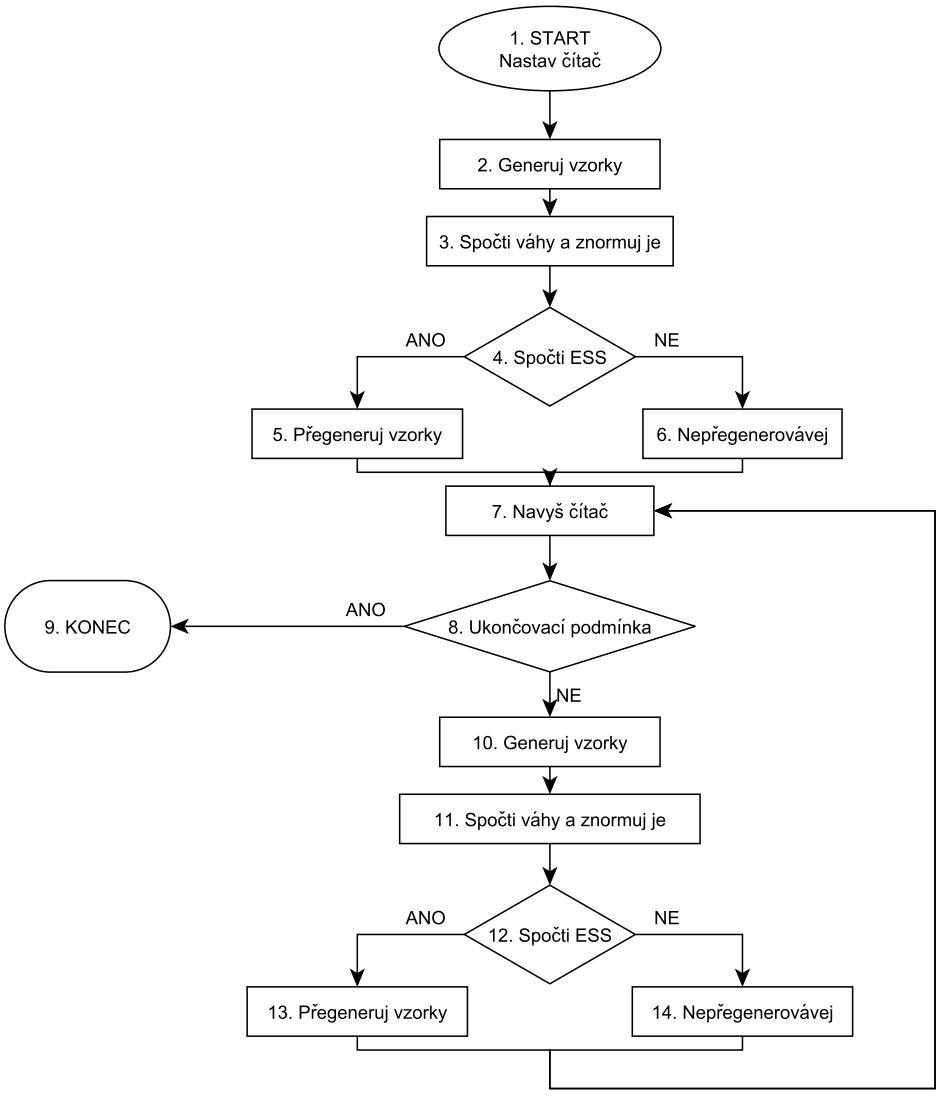
$$\alpha_n(x_{1:n}) = \alpha_n(x_{n-1:n}) = \frac{g(y_n|x_n)f(x_n|x_{n-1})}{q(x_n|y_n, x_{n-1})}.$$

V čase n máme tedy díky adaptivnímu přegenerovávání k dispozici odhady

$$\widetilde{p}(x_{1:n}|y_{1:n}) = \sum_{i=1}^{N} W_n^i \delta_{X_{1:n}^i}(x_{1:n}),$$

$$\widetilde{p}(y_n|y_{1:n-1}) = \sum_{i=1}^{N} W_{n-1}^i \alpha_n(X_{n-1:n}^i).$$

Celý algoritmus můžeme pomocí vývojového diagramu zakreslit takto:



kde

1. Start, nastav čítač $n := 1$.
2. Generuj vzorky $X_1^i \sim q(x_1|y_1)$.
3. Spočti váhy $w_1(X_1^i) = \frac{\mu(X_1^i)g(y_1|X_1^i)}{q(X_1^i|y_1)}$ a znormuj je: $W_1^i = \frac{w_1(X_1^i)}{\sum_{j=1}^N w_1(X_1^j)}$.
4. Spočti $ESS = \left(\sum_{i=1}^N (W_1^i)^2\right)^{-1}$. Je $ESS < N_T$?
5. Přegeneruj vzorky $\{W_1^i, X_1^i\} \leftarrow \left\{\frac{1}{N}, \overline{X}_1^i\right\}$.
6. Nepřegenerovávej $\{W_1^i, X_1^i\} \leftarrow \{W_1^i, X_1^i\}$.
7. Navyš čítač $n := n + 1$.
8. Ukončovací podmínka: Je $n > n_{max}$?
9. Konec.
10. Generuj vzorky $X_n^i \sim q(x_n|y_n, X_{n-1}^i)$ a polož $X_{1:n}^i \leftarrow \left(X_{1:n-1}^i, X_n^i\right)$.
11. Spočti přírůstkové váhy $\alpha_n(X_{n-1:n}^i) = \frac{g(y_n|X_n^i)f(X_n^i|X_{n-1}^i)}{q(X_n^i|y_n, X_{n-1}^i)}$ a znormuj je: $W_n^i = \frac{w_{n-1}(X_{1:n-1}^i)\alpha_n(X_{n-1:n}^i)}{\sum_{j=1}^N w_{n-1}(X_{1:n-1}^j)\alpha_n(X_{n-1:n}^j)}$.
12. Spočti $ESS = \left(\sum_{i=1}^N (W_n^i)^2\right)^{-1}$. Je $ESS < N_T$?
13. Přegeneruj vzorky $\{W_n^i, X_{1:n}^i\} \leftarrow \left\{\frac{1}{N}, \overline{X}_{1:n}^i\right\}$.
14. Nepřegenerovávej $\{W_n^i, X_{1:n}^i\} \leftarrow \{W_n^i, X_{1:n}^i\}$.

Kapitola 4

Simulace

V této kapitole se zaměříme na popis konkrétní realizace algoritmu provedené na nasimulovaných datech. Nejdříve popíšeme model a volbu jednotlivých funkcí, poté bude uveden zdrojový kód skriptu v jazyce R, kterým jsme získali prezentované výsledky. Na závěr uvedeme drobnou modifikaci, kdy přijmeme chybný předpoklad o rozdělení $g(y_n|X_n^i)$, což může způsobit problémy u reálných dat, a uvidíme, jak to ovlivní náš odhad.

4.1 Popis modelu

Neznámým procesem $\{X_n\}_{n\geq 1}$ je pro nás náhodná procházka s diskrétním časem a spojitou množinou stavů. Pozorování Y_n jsou z Poissonova rozdělení s parametry $\lambda_n = e^{X_n^i}$ pro $n \geq 1$. Náhodný proces se spojitou množinou stavů $\{X_n\}_{n\geq 1}$ tedy pozorujeme prostřednictvím procesu s diskrétní množinou stavů $\{Y_n\}_{n\geq 1}$.

- Za $\{X_n\}_{n\geq 1}$ je zvolena náhodná procházka s diskrétním časem a spojitou množinou stavů definovaná takto: $X_1 \sim \mathcal{N}(2; 0{,}2)$, $\{X_n \sim \mathcal{N}(X_{n-1}; 0{,}2)\}_{n\geq 2}$.
- Pozorování $\{Y_n\}_{n\geq 1}$ jsou z Poissonova rozdělení s parametry $\lambda_n = e^{X_n}$.
- První vzorky X_1^i generujeme z $\mathcal{N}(2; 0{,}2)$, protože je to předpoklad o rozdělení původního procesu $\{X_n\}_{n\geq 1}$. Pro jednoduchost vypouštíme závislost na y_1.
- Váhy poté spočteme dle vzorce $w_1(X_1^i) = \frac{\mu(X_1^i)g(y_1|X_1^i)}{q(X_1^i|y_1)}$, kde $\mu(X_1^i)$ se pokrátí s $q(X_1^i|y_1)$ díky zjednodušení s vypuštěním závislosti na y_1. Takže váhy počítáme jako $w_1(X_1^i) = g(y_1|X_1^i) = \frac{\lambda_1^{Y_1} e^{-\lambda_1}}{Y_1!}$, kde $\lambda_1 = e^{X_1^i}$.

- Normované váhy spočteme jako $W_1^i = \frac{w_1(X_1^i)}{\sum_{j=1}^N w_1(X_1^j)}$.
- V čase n vzorky X_n^i generujeme z $\mathcal{N}(X_{n-1}^i; 0{,}2)$.
- Přírůstkové váhy $\alpha_n(X_{n-1:n}^i)$ díky popsanému zjednodušení počítáme jako $\alpha_n(X_{n-1:n}^i) = g(y_n|X_n^i) = \frac{\lambda^{Y_n} e^{-\lambda}}{Y_n!}$, kde $\lambda = e^{X_n^i}$.
- Váhy spočteme jako $w_n(X_{1:n}^i) = w_{n-1}(X_{1:n-1}^i)\alpha_n(X_{n-1:n}^i)$, protože si pamatujeme váhy z minulého kroku.
- Normované váhy W_n^i poté spočteme jako $W_n^i = \frac{w_n(X_{1:n}^i)}{\sum_{j=1}^N w_n(X_{1:n}^j)}$.
- Pokud ESS klesne pod zadanou mez N_T, přegenerováváme.
- Odhad $\widetilde{p}_n(x_n|y_{1:n}) = \sum_{i=1}^N W_n^i \delta_{X_n^i}(x_n)$.

4.2 Realizace

Výše popsaný model realizujeme pomocí jazyka R. Na přiloženém CD je k dispozici zdrojový kód v souboru *pf.r*, který je možno spustit pomocí statistického softwaru R (`http://www.r-project.org/`). Získání totožných výsledků lze docílit nastavením generátoru pseudonáhodných čísel na hodnotu 1018 příkazem

```
set.seed(1018)
```

Následuje kompletní komentovaný zdrojový kód.

```
PREDPOKLAD <- 0

T <- 1000 # Čas -- odpovídá n_{max}
N <- 10000 # Počet vzorků
N.T <- N/2 # Tolerance pro ESS

# =========== INICIALIZACE PROCESU ===========

X <- numeric(length=T)
X <- rnorm(T,0,0.2)
X <- cumsum(X)+2 # Spočtení náhodné procházky, proces X_n
Y <- rpois(T,exp(X)) # Pozorování, proces Y_n
```

```
# =========== PARTICLE FILTER ===========

VZORKY <- numeric(length=N)
VZORKY.MINULE <- numeric(length=N)
VAHY <- numeric(length=N)
VAHY.MINULE <- numeric(length=N)
NORMVAHY <- numeric(length=N)
NORMVAHZVAHY.MINULE <- numeric(length=N)
PRIRUSTKOVEVAHY <- numeric(length=N)

ODHAD <- numeric(length=T) # Náš odhad procesu X_n
U <- numeric(length=N) # Pro přegenerovávání vzorků
# metodou systematického přegenerovávání

OFFSPRINGS <- numeric(length=N) # U přegenerovávání:
# Čísla potomků N_n^i
HRANICE <- numeric(length=N) # U přegenerovávání:
# Částečné součty W_n^k

CASY <- numeric(length=T) # Časy, kdy přegenerováváme
ESSka <- numeric(length=T) # Jak se vyvíjí ESS

Pregeneruj <- function() { # Funkce pro přegenerovávání vzorků
  U[1] <- runif(1, min=0, max=(1/N))
  for(i in 2:N) {
    U[i] <- U[i-1] + (1/N)
  }
  HRANICE <- cumsum(NORMVAHY) # Spočtení částečných
  # součtů normovaných vah W_n^k

  OFFSPRINGS <- numeric(length=N) # Vynulování čísel potomků
  # využíváme faktu, že U_i jsou vzestupně seřazené, různé
```

```
# ----- SPOČTENÍ ČÍSEL POTOMKŮ -----
  j <- 1;  jm <- 1
  while(j<=N) {
    if(U[j] <= HRANICE[1]) {
      OFFSPRINGS[1] <- OFFSPRINGS[1] + 1
      jm <- jm + 1
    } else {
      j <- N # Vyskočení z cyklu
    }
    j <- j+1
  }
  for(i in 2:N) {
    j <- jm
    while(j<=N) {
      if(U[j] <= HRANICE[i]) {
        if(U[j] >= HRANICE[i-1]) {
          OFFSPRINGS[i] <- OFFSPRINGS[i] + 1
          jm <- jm + 1
        }
      } else {
        j <- N # Vyskočení z cyklu
      }
      j <- j+1
    }
  }
# ----- SPOČTENÍ PŘEGENEROVANÝCH VZORKŮ -----
  PREGENER <- numeric(length=N) # Přegenerované vzorky
  k <- 1
  for(i in 1:N) {
    if(OFFSPRINGS[i] != 0) {
      for(j in k:(k+OFFSPRINGS[i]-1)) {
        PREGENER[j] <- VZORKY[i]
      }
      k <- k+OFFSPRINGS[i]
    }
  }
  return(PREGENER)
}
```

```
# ----- ZAČÁTEK ALGORITMU -----

VZORKY <- rnorm(N,2,0.2) # Vygenerování prvních vzorků
if(PREDPOKLAD == 0) {
  VAHY <- dpois(Y[1], lambda=exp(VZORKY)) # Spočtení
  # vah w_1 se správným předpokladem
} else {
  VAHY <- dnorm(Y[1], exp(VZORKY), 0.5) # Spočtení
  # vah w_1 se špatným předpokladem
}
# --- Normování ---
soucet.vah <- sum(VAHY)
NORMVAHY <- (VAHY/soucet.vah)
# --- Spočtení ESS + podmínka ---
ESS <- (sum(NORMVAHY^2))^(-1)
if(ESS < N.T) { # Podmínka přegenerování
  VZORKY <- Pregeneruj() # Přegenerování
  VAHY <- 1/N
  NORMVAHY <- 1/N
  CASY[1] <- 1
}
ESSka[1] <- ESS
ODHAD[1] <- sum(NORMVAHY*VZORKY) # Spočtení prvního odhadu
# --- Cyklus podle času ---
for(n in 2:T) {
  VZORKY.MINULE <- VZORKY
  VAHY.MINULE <- VAHY
  NORMVAHY.MINULE <- NORMVAHY
  VZORKY <- rnorm(N,VZORKY.MINULE,0.2) # Generování vzorků
  if(PREDPOKLAD == 0) {
    PRIRUSTKOVEVAHY <- dpois(Y[n], lambda=exp(VZORKY))
    # Spočtení přírůstkových vah alpha_n se správným předpokladem
  } else {
    PRIRUSTKOVEVAHY <- dnorm(Y[n], exp(VZORKY), 0.5)
    # Spočtení přírůstkových vah alpha_n se špatným předpokladem
  }
  VAHY <- VAHY.MINULE*PRIRUSTKOVEVAHY # Spočtení vah w_n
```

```
  # --- Normování + ESS ---
  soucet.vah <- sum(VAHY)
  if(soucet.vah != 0) {
    NORMVAHY <- (VAHY/soucet.vah)
    ESS <- (sum(NORMVAHY^2))^(-1)
  } else {
    ESS <- 0
  }
  ESSka[n] <- ESS
  if(ESS < N.T) { # Podmínka přegenerování
    VZORKY <- Pregeneruj()
    VAHY <- 1/N
    NORMVAHY <- 1/N
    CASY[n] <- 1
  }
  ODHAD[n] <- sum(NORMVAHY*VZORKY) # Spočtení odhadu v čase n
}
```

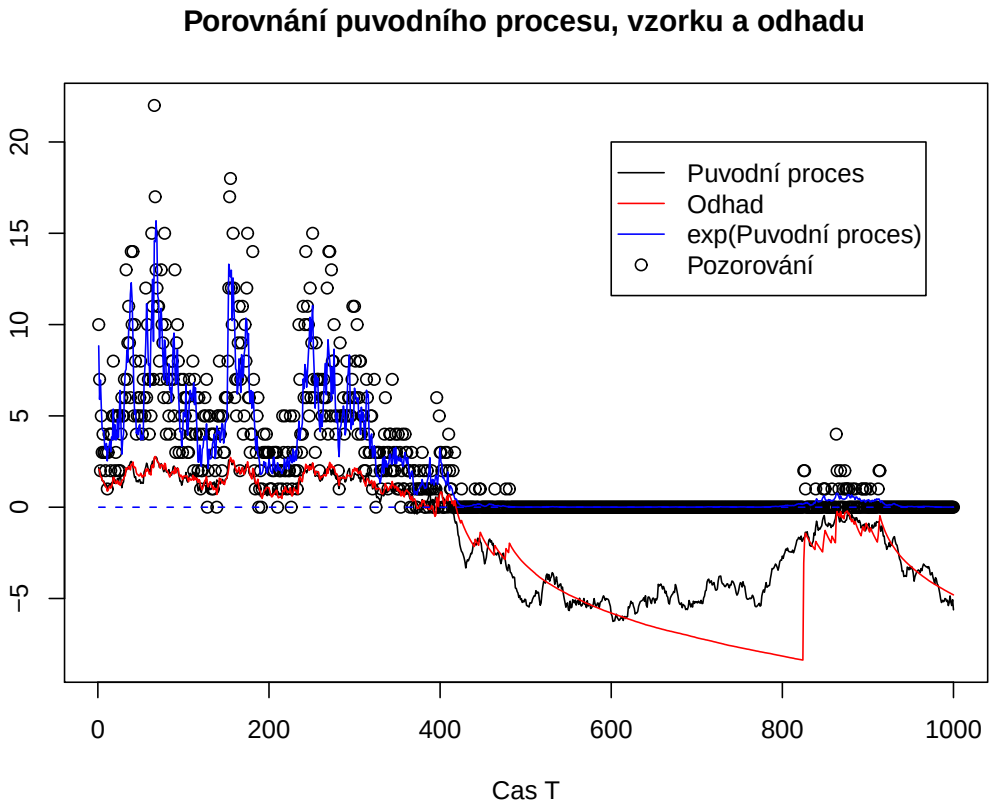
Časová složitost algoritmu je $\mathcal{O}(Nn)$ a paměťová $\mathcal{O}(N+n)$, takže pro naše data, při volbě $n = 1000$ a $N = 10$ nebo $N = 10000$, je algoritmus na běžném počítači svižný a výpočet trvá nanejvýš několik vteřin.

4.3 Výstupy

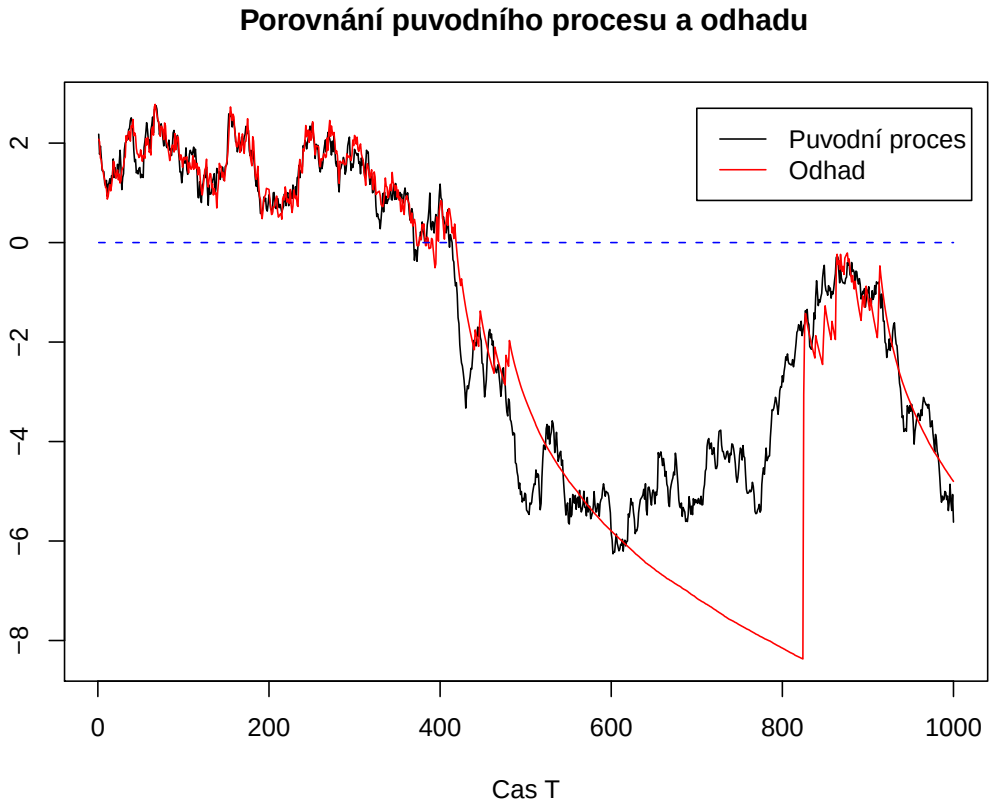
Jak už jsme nastínili, algoritmus jsme odzkoušeli na nasimulovaných datech při volbě $n = 1000$ a $N = 10$ nebo $N = 10000$.

Na prvním obrázku 4.1 jsou zobrazeny dohromady jak původní proces $\{X_n\}_{n\geq 1}$ (černá linka), tak pozorování $\{Y_n\}_{n\geq 1}$ (černé body). Modrou linkou jsou pospojovány hodnoty e^{X_n}. Jedná se o hodnoty λ_n, z kterých jsou poté Poissonovsky získána naše pozorování. Červenou linkou je zobrazen náš odhad $\widetilde{p}_n(x_n|y_{1:n})$ procesu $\{X_n\}_{n\geq 1}$. Obrázek je pořízen při volbě $N = 10000$. Důsledkem toho je, že pokud odhadovaný proces $\{X_n\}_{n\geq 1}$ začne nabývat hodnoty menší než 0, $\lambda_n = e^{X_n}$ budou z malého intervalu blízkého 0. Pozorujeme, jak začne být odhad $\widetilde{p}_n(x_n|y_{1:n})$ původního procesu $\{X_n\}_{n\geq 1}$ nepřesný i při volbě vyššího N. Tento jev je způsoben tím, že v pozorováních $\{Y_n\}_{n\geq 1}$ není prostě dostatek informace o průběhu původního procesu $\{X_n\}_{n\geq 1}$.

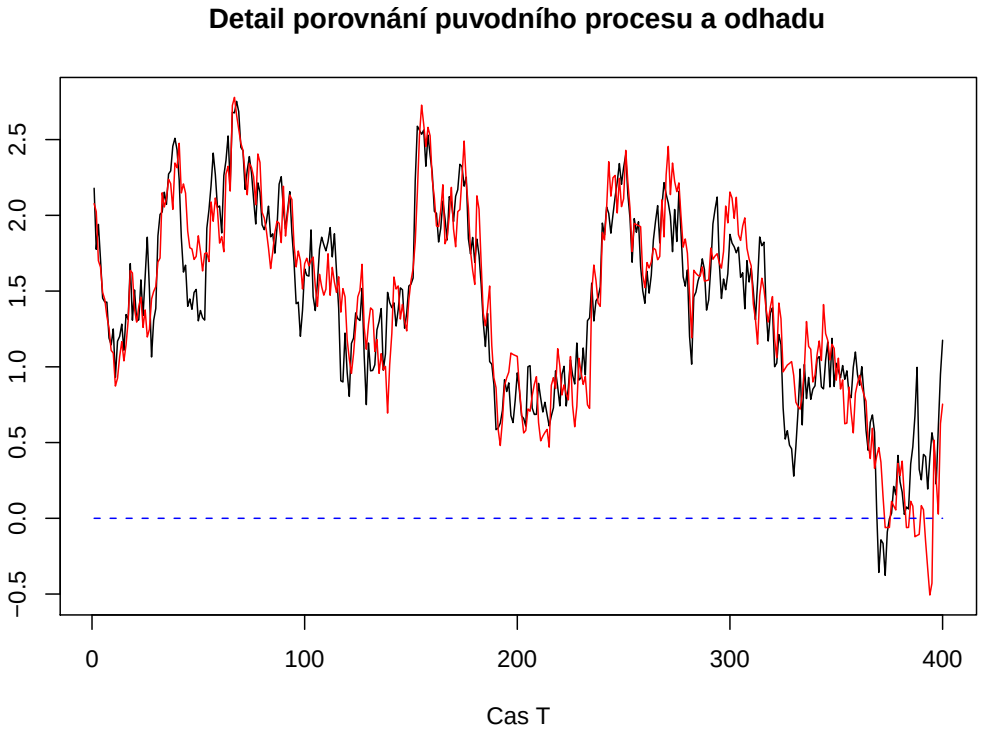
Na druhém grafu 4.2 jsou pro přehlednost vyobrazeny pouze původní proces $\{X_n\}_{n\geq 1}$ a náš odhad $\widetilde{p}_n(x_n|y_{1:n})$ při volbě $N = 10000$.



Obrázek 4.1: Situace při $N = 10000$

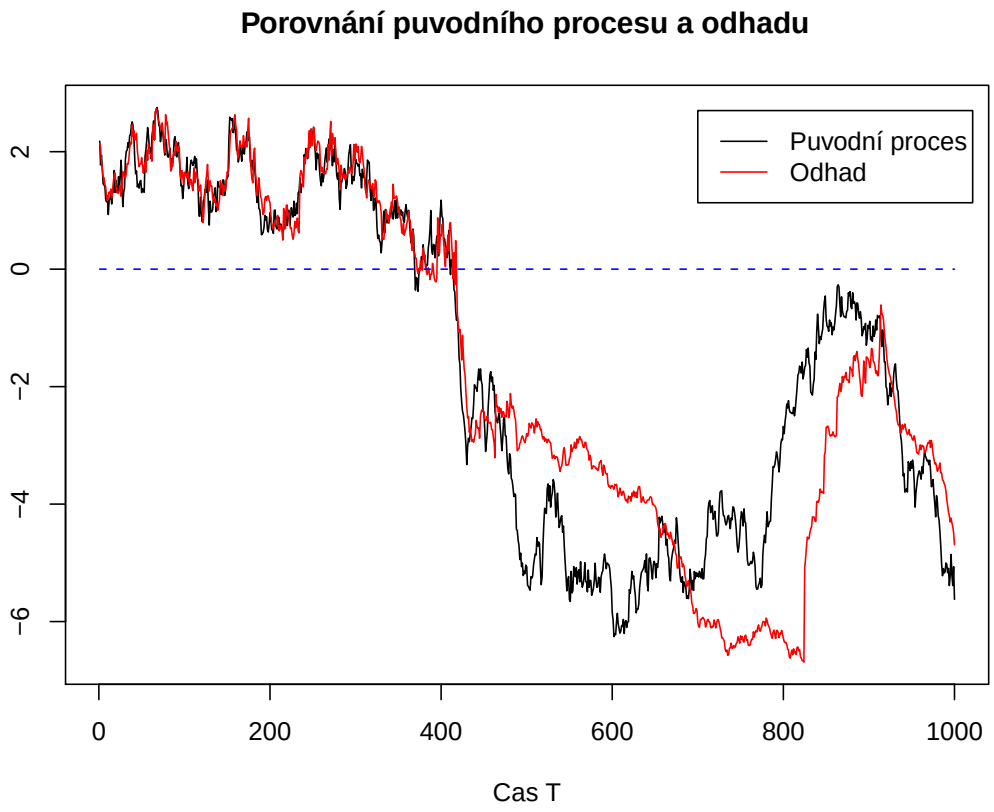


Obrázek 4.2: Porovnání $\{X_n\}_{n\geq 1}$ s odhadem $\widetilde{p}_n(x_n|y_{1:n})$ při $N = 10000$



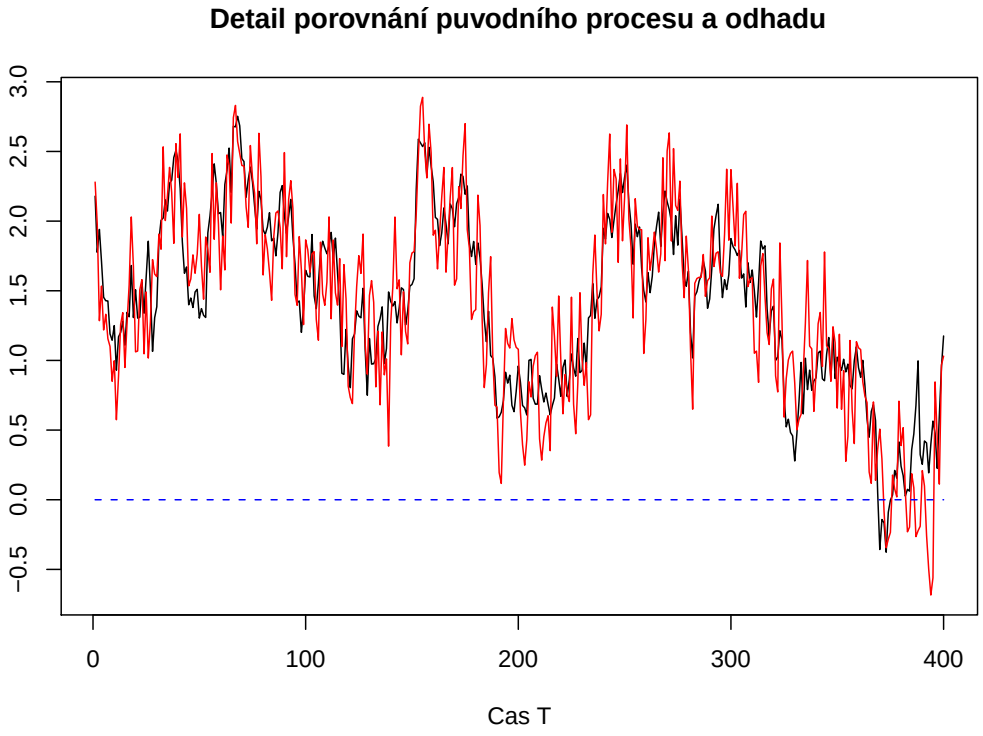
Obrázek 4.3: Detail porovnání $\{X_n\}_{n\geq 1}$ s odhadem $\widetilde{p}_n(x_n|y_{1:n})$ při $N = 10000$ jen do času $n = 400$

Na obrázku 4.3 je k vidění detail porovnání $\{X_n\}_{n\geq 1}$ s naším odhadem $\widetilde{p}_n(x_n|y_{1:n})$ do času $n = 400$, tedy přibližně do chvíle, kdy je v pozorováních ještě dostatek informace k provedení rozumného odhadu.



Obrázek 4.4: Porovnání $\{X_n\}_{n\geq 1}$ s odhadem při $N = 10$

Na dalším obrázku 4.4 je porovnání procesu $\{X_n\}_{n\geq 1}$ s naším odhadem $\widetilde{p}_n(x_n|y_{1:n})$ při volbě $N = 10$. Vidíme, že už při tak malé volbě dosahujeme uspokojivých výsledků, které ani zvýšení počtu vzorků na $N = 10000$ nevylepší zásadním způsobem.



Obrázek 4.5: Detail porovnání $\{X_n\}_{n\geq 1}$ s odhadem $\widetilde{p}_n(x_n|y_{1:n})$ při chybném předpokladu o pozorováních při $N = 10000$ jen do času $n = 400$

Na posledním obrázku 4.5 je vyobrazena situace, kdy přijmeme o rozdělení $g(y_n|X_n^i)$ chybný předpoklad při $N = 10000$. Tedy místo správného předpokladu, že $g(y_n|X_n^i)$ je z Poissonova rozdělení s parametry $\lambda_n = e^{X_n^i}$, přijímáme, že $g(y_n|X_n^i) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(y_n-\mu_n)^2}{2\sigma^2}}$, kde $\mu_n = e^{X_n^i}$ a $\sigma^2 = 0{,}5$.

- Pozorování $\{Y_n\}_{n\geq 1}$ jsou z normálního rozdělení $\mathcal{N}(X_n; 0{,}5)$.
- První váhy tedy spočteme jako $w_1(X_1^i) = g(y_1|X_1^i) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(y_1-\mu_1)^2}{2\sigma^2}}$, kde $\mu_1 = e^{X_1^i}$ a $\sigma^2 = 0{,}5$.
- Přírůstkové váhy $\alpha_n(X_{n-1:n}^i)$ počítáme jako $\alpha_n(X_{n-1:n}^i) = g(y_n|X_n^i) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(y_n-\mu)^2}{2\sigma^2}}$, kde $\mu = e^{X_n^i}$ a $\sigma^2 = 0{,}5$.

- Normované váhy W_n^i poté spočteme jako $W_n^i = \frac{w_{n-1}(X_{1:n-1}^i)\alpha_n(X_{n-1:n}^i)}{\sum_{j=1}^N w_{n-1}(X_{1:n-1}^j)\alpha_n(X_{n-1:n}^j)}$, protože si pamatujeme váhy z minulého kroku.

Při výpočtu s tímto chybným předpokladem algoritmus přegenerovává přibližně 360 krát, kdežto se správným jen asi 138 krát. Porovnáním obrázků 4.4 a 4.5 vidíme, že přijetím špatného předpokladu se stává odhad méně vypovídajícím.

Závěr

V praxi se setkáváme velmi často s problémem odhadování neznámých veličin z nepřesných nebo nějakým způsobem pokřivených pozorování. V mnoha těchto případech máme určité informace, jak mají tato data vypadat. To nás vede k formulování Bayesovského modelu, kde odhadované pravděpodobnosti závisí na pozorování. Chceme-li navíc zpracovávat přicházející data v reálném čase, Particle filtering je mocným nástrojem, který na principu simulace dokáže tyto problémy řešit.

V této práci jsme shrnuli základní teoretické výsledky, sestavili algoritmus a odzkoušeli jej na nasimulovaných datech. Jelikož se jednalo o data uměle nagenerovaná, věděli jsme o nich explicitně z jakého jsou rozdělení, a to celý problém zjednodušovalo. Použití našeho algoritmu na reálná data by bylo o poznání komplikovanější, jelikož bychom museli učinit další dodatečné předpoklady o struktuře odhadovaných dat. Toto je směr, kterým by se dala tato práce dále rozšiřovat.

Ať už se tedy jedná (viz [4]) o finanční modely v ekonometrii, analýzu časových řad, navigaci v terénu, neuronové sítě, předpověď počasí, robotiku, strojové učení, rozpoznávání objektů v obraze, průmyslovou kontrolu jakosti, modelování biologických populací nebo o navádění raket ve zbrojním průmyslu, zjišťujeme, že metody popsané v této práci se v komplikovanějších modifikacích objevují napříč celou lidskou činností.

Literatura

[1] Pawlas, Z.: Metody MCMC (STP139). `http://www.karlin.mff.cuni.cz/~pawlas/2007/STP139/mcmc.pdf`

[2] Lachout P., Prášková, Z. (2005): Základy náhodných procesů. Karolinum, Praha.

[3] Doucet, A., Johansen, A. M. (2008): A Tutorial on Particle Filtering and Smoothing: Fifteen years later.

[4] Doucet, A., de Freitas, N., Gordon, N.J. (2001): An introduction to sequential Monte Carlo methods. in *Sequential Monte Carlo Methods in Practise*. Springer-Verlag, New York.